SoMachine
TimeSync
Library Guide

06/2017

EIO0000002542.00

www.schneider-electric.com

Schneider Electric

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

No part of this document may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without express written permission of Schneider Electric.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

Table of Contents

Safety Information

Important Information

NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

The addition of this symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.

This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

⚠ DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

⚠ WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

⚠ CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result** in minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

BEFORE YOU BEGIN

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

⚠ WARNING

UNGUARDED EQUIPMENT

- Do not use this software and related automation equipment on equipment which does not have point-of-operation protection.
- Do not reach into machinery during operation.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.

Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.

In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.

Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

START-UP AND TEST

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check be made and that enough time is allowed to perform complete and satisfactory testing.

⚠ WARNING

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and temporary grounds that are not installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove all temporary grounds from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

OPERATION AND ADJUSTMENTS

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

About the Book

At a Glance

Document Scope

This document describes the library TimeSync.

The TimeSync library implements the SNTP (Simple Network Time Protocol) client feature. It allows your controller to connect to an NTP (Network Time Protocol) or SNTP time server in order to synchronize the internal RTC (Real-Time Clock) of the controller in accordance with the primary time standard UTC (Universal Time Coordinated) that is globally unique.

The TimeSync library uses system functions and resources which are supported on specific controller platforms:

- Modicon M241 Logic Controller
- Modicon M251 Logic Controller
- Modicon M258 Logic Controller
- Modicon LMC078 Motion Controller
- Modicon LMC058 Motion Controller

Validity Note

This document has been updated for the release of SoMachine V4.3.

The technical characteristics of the devices described in this document also appear online. To access this information online:

Step	Action
1	Go to the Schneider Electric home page *www.schneider-electric.com*.
2	In the **Search** box type the reference of a product or the name of a product range. ● Do not include blank spaces in the reference or product range. ● To get information on grouping similar modules, use asterisks (*).
3	If you entered a reference, go to the **Product Datasheets** search results and click on the reference that interests you. If you entered the name of a product range, go to the **Product Ranges** search results and click on the product range that interests you.
4	If more than one reference appears in the **Products** search results, click on the reference that interests you.
5	Depending on the size of your screen, you may need to scroll down to see the data sheet.
6	To save or print a data sheet as a .pdf file, click **Download XXX product datasheet**.

The characteristics that are presented in this manual should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the manual and online information, use the online information as your reference.

Related Documents

Document title	Reference
SoMachine Functions and Libraries User Guide	*EIO0000000735 (ENG)*; *EIO0000000792 (FRE)*; *EIO0000000793 (GER)*; *EIO0000000795 (SPA)*; *EIO0000000794 (ITA)*; *EIO0000000796 (CHS)*
SoMachine Programming Guide	*EIO0000000067 (ENG)*; *EIO0000000069 (FRE)*; *EIO0000000068 (GER)*; *EIO0000000071 (SPA)*; *EIO0000000070 (ITA)*; *EIO0000000072 (CHS)*
SoMachine Getting & Setting Real Time Clock SysTimeRtc and SysTimeCore Library Guide	*EIO0000000667 (ENG)*; *EIO0000000668 (FRE)*; *EIO0000000669 (GER)*; *EIO0000000671 (SPA)*; *EIO0000000670 (ITA)*; *EIO0000000672 (CHS)*
SoMachine TcpUdpCommunication Library Guide	*EIO0000002204 (ENG)*; *EIO0000002255 (FRE)*; *EIO0000002205 (GER)*; *EIO0000002257 (SPA)*; *EIO0000002256 (ITA)*; *EIO0000002258 (CHS)*
SoMachine Example Guide - RTC Control / Timezone / Daylight Saving Time	*EIO0000002445 (ENG)*

You can download these technical publications and other technical information from our website at *http://www.schneider-electric.com/en/download*.

Part I
General Information

What Is in This Part?

This part contains the following chapters:

Chapter 1
Specific Safety Information

Overview

This section contains information regarding working with the TimeSync library. Personnel working with the TimeSync library must read and observe this information.

What Is in This Chapter?

This chapter contains the following topics:

Qualification of Personnel

Overview

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel.

No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and the installation, and has received safety-related training to recognize and avoid the hazards involved.

The qualified person must be able to detect possible hazards that may arise from parameterization, modifying parameter values and generally from mechanical, electrical, or electronic equipment. The qualified person must be familiar with the standards, provisions, and regulations for the prevention of industrial accidents, which they must observe when designing and implementing the system.

Proper Use

Overview

This product is a library to be used together with the control systems and servo amplifiers intended solely for the purposes described in the present documentation as applied in the industrial automation sector.

Always observe the applicable safety-related instructions, the specified conditions, and the technical data.

Perform a risk evaluation concerning the specific use before using the product. Take protective measures according to the result.

Since the product is used as a part of an overall system, you must ensure the safety of the personnel by means of the concept of this overall system (for example, machine concept).

Any other use is not intended and may be hazardous. Electrical devices and equipment must only be installed, operated, maintained, and repaired by qualified personnel.

Product Related Information

Product Related Information

⚠ WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.[1]
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

[1] For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

Before you attempt to provide a solution (machine or process) for a specific application using the POUs found in the library, you must consider, conduct and complete best practices. These practices include, but are not limited to, risk analysis, functional safety, component compatibility, testing and system validation as they relate to this library.

⚠ WARNING

IMPROPER USE OF POUS

- Perform a safety-related analysis for the application and the devices installed.
- Ensure that the POUs are compatible with the devices in the system and have no unintended effects on the proper functioning of the system.
- Use appropriate parameters, especially limit values, and observe machine wear and stop behavior.
- Verify that the sensors and actuators are compatible with the selected POUs.
- Thoroughly test all functions during verification and commissioning in all operation modes.
- Provide independent methods for critical control functions (emergency stop, conditions for limit values being exceeded, etc.) according to a safety-related analysis, respective rules, and regulations.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Update your application program as required, paying particular attention to I/O address adjustments, whenever you modify the hardware configuration.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Incomplete file transfers, such as data files, application files and/or firmware files, may have serious consequences for your machine or controller. If you remove power, or if there is a power outage or communication interruption during a file transfer, your machine may become inoperative, or your application may attempt to operate on a corrupted data file. If a an interruption occurs, reattempt the transfer. Be sure to include in your risk analysis the impact of corrupted data files.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION, DATA LOSS, OR FILE CORRUPTION

- Do not interrupt an ongoing data transfer.
- If the transfer is interrupted for any reason, re-initiate the transfer.
- Do not place your machine into service until the file transfer has completed successfully, unless you have accounted for corrupted files in your risk analysis and have taken appropriate steps to prevent any potentially serious consequences due to unsuccessful file transfers.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Chapter 2
Presentation of the Library

General Information

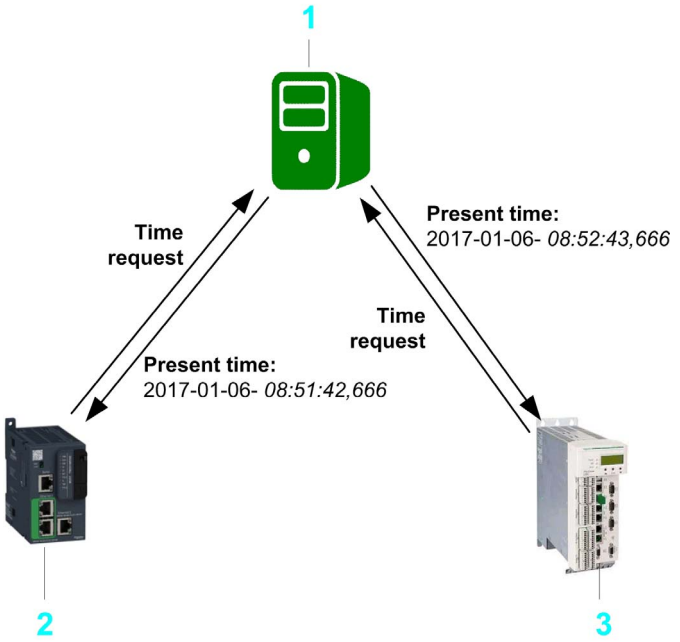
Introduction

The TimeSync library implements the SNTP (Simple Network Time Protocol) client feature. It allows your controller to connect to an NTP (Network Time Protocol) or SNTP time server in order to synchronize the internal RTC (Real-Time Clock) of the controller in accordance with the primary time standard UTC (Universal Time Coordinated) that is globally unique.

The SNTP client complies to version 4 of the SNTP protocol.

It provides the following functions:

- Generating a request to an (S)NTP server
- Receiving and structuring the response of the (S)NTP server
- Calculating the round-trip delay and the clock offset
- Providing the synchronized time stamp for synchronizing the RTC of the controller and taking the offset between local RTC and (S)NTP server time as well as round-trip delay times into account
- Managing detected errors



1 Time server: SNTP or NTP server
2 Controller X: SNTP client
3 Controller Y: SNTP client

The following table indicates the characteristics of the library:

Characteristic	Value
Library title	TimeSync
Company	Schneider Electric
Category	**Communication**
Default namespace	**TIMS**
Language model attribute	Qualified-access-only *(see SoMachine, Functions and Libraries User Guide)*
Forward compatible library	Yes (FCL *(see SoMachine, Functions and Libraries User Guide)*)

NOTE: For this library, qualified-access-only is set. This means, that the POUs, data structures, enumerations, and constants have to be accessed using the namespace of the library. The default namespace of the library is **TIMS**.

General Considerations

NOTE: Setting the RTC of the controller generates entries into the controller log file. Therefore, for automatic adjustments, do not use this function more than once a day.

The allowed value range in which the internal RTC (Real-Time Clock) of the controller can be set by the function block is restricted to 01.01.1970...01.01.2100. The accuracy depends on the (S)NTP server that is used. Maximum accuracy of 1 s can be reached for UTC synchronization.

NOTE: The entries in the controller log file are saved with a time stamp. Be aware that when setting the internal RTC (Real-Time Clock) of the controller back in time, that this can result in two different log entries that are saved with an apparently inappropriate time stamp.

Consider the following limitations for (S)NTP communication:
- Only IPv4 (Internet Protocol version 4) is supported.
- Only one request to one (S)NTP time server is allowed at a time.
- The TimeSync library incorporates pointers on addresses.
- Do not use a poll interval less than 15 seconds.
- Use local servers whenever available to avoid unnecessary traffic on backbone network.

Executing the **Online Change** command can change the contents of addresses.

⚠ CAUTION

INVALID POINTER

Verify the validity of the pointers when using pointers on addresses and executing the Online Change command.

Failure to follow these instructions can result in injury or equipment damage.

The library described in this document internally uses the TcpUdpCommunication library.

The TcpUdpCommunication (Schneider Electric) and the CAA Net Base Services library (CAA Technical Workgroup) use the same system resources on the controller. The simultaneous use of both libraries in the same application may lead to disturbances during the operation of the controller.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not use the library TcpUdpCommunication (Schneider Electric) together with the library CAA Net Base Services (CAA Technical Workgroup) simultaneously in the same application.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Considerations Concerning Cyber Security

The TimeSync library functions do not support secure connections such as TLS (Transport Layer Security) or SSL (Secure Socket Layer). Since the SNTP telegrams are not encrypted and authentication is not required to get or set information on an agent. Communication must only be performed inside your industrial network, isolated from other networks inside your company, and protected from the Internet.

NOTE: Schneider Electric adheres to industry best practices in the development and implementation of control systems. This includes a "Defense-in-Depth" approach to secure an Industrial Control System. This approach places the controllers behind one or more firewalls to restrict access to authorized personnel and protocols only.

⚠ WARNING

UNAUTHENTICATED ACCESS AND SUBSEQUENT UNAUTHORIZED MACHINE OPERATION

- Evaluate whether your environment or your machines are connected to your critical infrastructure and, if so, take appropriate steps in terms of prevention, based on Defense-in-Depth, before connecting the automation system to any network.
- Limit the number of devices connected to a network to the minimum necessary.
- Isolate your industrial network from other networks inside your company.
- Protect any network against unintended access by using firewalls, VPN, or other, proven security measures.
- Monitor activities within your systems.
- Prevent subject devices from direct access or direct link by unauthorized parties or unauthenticated actions.
- Prepare a recovery plan including backup of your system and process information.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

For more information on organizational measures and rules covering access to infrastructures, refer to ISO/IEC 27000 series, Common Criteria for Information Technology Security Evaluation, ISO/IEC 15408, IEC 62351, ISA/IEC 62443, NIST Cybersecurity Framework, Information Security Forum - Standard of Good Practice for Information Security.

The library incorporates the following features that help to prevent communication errors:
- Verification of the time stamp that has been sent originally to the server to help to detect transmission errors.
- Verification that the definition of the protocol is in accordance with the standard.
- Verification of the server version and issuing a diagnostic message in case of mismatch.

Overview of the POUs

Function block / function	Use
`FB_SntpClient` *(see page 41)*	Provides SNTP client functions.
`FC_EtResultToString` *(see page 45)*	Converts an enumeration element of type `ET_Result` to a string value.

Overview of the Structures in the Module-Specific Interface

Structure	Use
`ST_TimeRequest` *(see page 32)*	Contains specific information for sending an SNTP request to an (S)NTP time server.
`ST_TimeResponse` *(see page 33)*	Contains the time stamps and can be used to synchronize the RTC.

Overview of the Enumerations

Enumeration	Use
`ET_ComMode` *(see page 28)*	Defines the communication mode of the SNTP client.
`ET_Result` *(see page 29)*	Contains the possible values that indicate the result of operations executed by the function block.

Part II
Enumerations and Structures

What Is in This Part?

This part contains the following chapters:

Chapter 3
Enumerations

What Is in This Chapter?

This chapter contains the following topics:

ET_ComMode

Overview

Type:	Enumeration
Available as of:	V1.0.7.0

Description

The enumeration ET_ComMode defines the communication mode of the SNTP client. This version of the library only supports unicast mode. Therefore, the value is fixed to 0.

Enumeration Elements

Name	Data type	Value	Description
Unicast	UDINT	0	A point-to-point connection is established: A unicast client sends a request to a designated server at its unicast address and expects a reply from which it can determine the time.

Used By

- ST_TimeRequest

ET_Result

Overview

Type:	Enumeration
Available as of:	V1.0.7.0

Description

The enumeration ET_Result contains the possible values that indicate the result and extended error codes of operations executed by the function block.

Enumeration Elements

q_xError of the function block is TRUE as long as these results are set:

Name	Data type	Value	Description
The following status messages are shown if no error has been detected:			
Ok	UDINT	0	The operation was completed successfully and the function block is idle.
Initializing	UDINT	1	The function block is initializing.
UDPopen	UDINT	2	The function block opens a UDP socket.
UDPsend	UDINT	3	The function block uses the input data to build and send a request telegram.
Processing	UDINT	4	The function block is processing the received telegram.
Disabled	UDINT	5	The function block is disabled.
The following status messages are shown if an error has been detected in the input data:			
InvalidServerIp	UDINT	21	It has been detected that the value of i_sServerIp is not valid. The IP address defined by this input *(see page 32)* must consist of 4 numbers in the range of 0...255 separated by periods.
InputTimeoutZero	UDINT	22	It has been detected that the value of i_timServerTimeOut is not valid *(see page 32)*. It must be greater than zero because it specifies the time the function block waits for a response from the server.
InvalidMaxRtcOffset	UDINT	23	It has been detected that the value of q_liRtcOffset *(see page 33)* is greater than 7200 (seconds).
InvalidComMode	UDINT	24	It has been detected that the value of ET_ComMode is not valid. Verify the settings enumeration elements for ET_ComMode *(see page 28)*.

Name	Data type	Value	Description
`InvalidTimeZone`	UDINT	25	It has been detected that the time zone offset values are greater than 86400 (seconds) (24 hours).
The following status messages are shown if an error has been detected in the received data:			
`TimeStampInvalid`	UDINT	30	It has been detected that the time stamp that has been sent to the time server for verification has not been returned. Make another attempt to connect to the time server or try to connect to another time server with a different IP address.
`ClockNotSynchronized`	UDINT	31	The clock of the time server has not been synchronized. Connect to an alternative time server.
`VersionNumberInvalid`	UDINT	32	It has been detected that the version of the time server is invalid. Verify that the version of the time server is either 3 or 4.
`ProtocolModeInvalid`	UDINT	33	It has been detected that the protocol mode of the time server is invalid. Verify that the protocol mode used by the time server is either 4 or 5.
`KodPacketReceived`	UDINT	34	An (S)NTP protection protocol packet has been received. Transmission of data to this server is stopped. Connect to an alternative time server.
`NewRtcValueOutOfRange`	UDINT	35	The new RTC value that has been calculated is not within the range of the present value ± input `i_uiMaxRtcOffset` *(see page 43)*.
The following status messages are shown if an error has been detected in UDP communications:			
`UdpIssue`	UDINT	40	A generic error has been received from the TcpUdpCommunication library. Refer to the TcpUdpCommunication Library Guide *(see SoMachine, TcpUdpCommunication, Library Guide)*.
`UdpTimeout`	UDINT	41	No response received from the agent in the given time with the given retries. You can try to increase `GPL.Gc_udiTimeout` and / or `GPL.Gc_uiMaxNumberRetries`.
`UnknownResult`	UDINT	99	Feedback of `FC_EtResultToString` in case of an undeterminable `EtResult` value.

In case of recurring timeouts and other communication issues:

- Verify physical connections of controller, SNTP agent, and so on.
- Verify communication route between controller and SNTP agent, for example, switch and router settings, firewall settings (UDP communication and SNTP ports need to be allowed).
- Verify provided IP address and SNTP port of the (S)NTP time server.

Used By

- `FB_SntpClient`

Chapter 4
Structures

What Is in This Chapter?

This chapter contains the following topics:

ST_TimeRequest

Overview

Type:	Structure
Available as of:	V1.0.7.0
Inherits from:	–

Description

The structure `ST_TimeRequest` contains specific information for sending an SNTP request to an (S)NTP time server.

Structure Elements

Name	Data type	Description
`i_sServerIp`	STRING[15]	IP address of the (S)NTP time server. It must be unique address that consists of four numbers in the range of 0...255 separated by periods. The IP address has to be unique in the network.
`i_uiServerPort`	UINT	SNTP port of the time server. If the input is empty, the default port 123 is used.
`i_timServerTimeOut`	TIME	Maximum time to wait for a response from the server. The value must be greater than zero.
`i_etComMode`	`ET_ComMode`	The communication mode defined with the `ET_ComMode` enumeration *(see page 28)*.

Used By

- `FB_SntpClient`

ST_TimeResponse

Overview

Type:	Structure
Available as of:	V1.0.7.0
Inherits from:	–

Description

The structure ST_TimeResponse contains the synchronized time stamp including the offset i_diTimeZone defined for FB_SntpClient *(see page 41)*. It can be used to synchronize the RTC of the controller and takes the received time information concerning time server offset and round-trip delay into account.

Structure Elements

Name	Data type	Description
q_stSyncTimestamp	SysTimeRtc.RTS_SYSTIMEDATE	Contains the synchronized time stamp (including the offset i_diTimeZone) presented in a readable format. Structure provided by the SysTimeRtc library *(see SoMachine, Getting & Setting Real Time Clock, SysTimeRtc and SysTimeCore Library Guide)*.
q_uliUnixTimestamp	ULINT	Synchronized time stamp (including the offset i_diTimeZone) as UNIX time stamp.
q_liRtcOffset	LINT	Calculated clock offset (in ms) between local RTC of the controller and the (S)NTP time server. Negative values are possible, for example -2 h = -7200000.
q_udiRoundTripDelay	UDINT	Calculated round-trip delay (in ms). Example: 30 = 30 ms

Used By

- FB_SntpClient

Part III
Global Variables

Chapter 5
Global Constants List

Global Constants List (GCL)

Overview

Type:	Global constants
Available as of:	V1.0.7.0

Description

The global constants list contains the global constants of the TimeSync library.

Global Constants

Variable	Data type	Value	Description
`Gc_sLibraryVersion`	STRING[80]	Vx.x.x.0[1]	Library version
[1] This value varies to indicate the version of the library.			

Part IV
Program Organization Units (POU)

What Is in This Part?

This part contains the following chapters:

Chapter 6
Function Blocks

FB_SntpClient

Overview

Type:	Function block
Available as of:	V1.0.7.0

Task

The FB_SntpClient function block provides SNTP client functions.

Functional Description

The FB_SntpClient function block is the user interface for SNTP communications.

The function block needs to be enabled and ready to perform a request. When starting the execution of a request, the information from the inputs is used to build an SNTP telegram containing the time request. The telegram is sent to a single server using UDP.

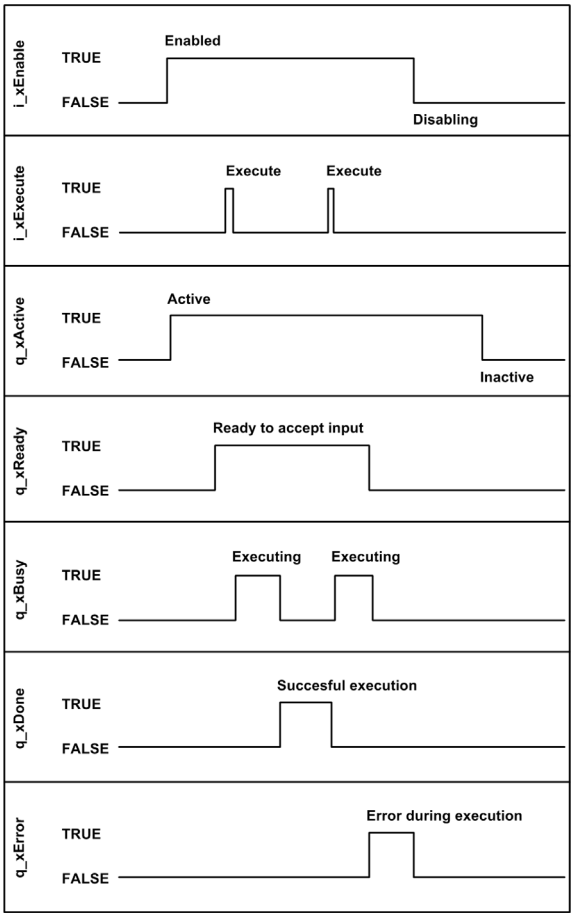
The function block waits for a response from the server, processes it and presents the received information at the output q_stTimeResponse. As long as the function block is executing a request, the output q_xBusy is set to TRUE and q_etResult presents the state of operation. The output q_xDone indicates a successful execution and q_xError shows if the function block detects an error during execution with q_etResult and q_sResultMsg presenting further information on the nature or cause of the detected error. If an error is detected, the function block needs to be reset by disabling and re-enabling it.

Depending on the input `i_xSyncRtc`, the calculated values are used to set the RTC of the controller to the value that has been received from the server (including the offset defined with the input `i_diTimeZone`). A maximum accuracy of 1 s can be reached for UTC synchronization.

NOTE: Daylight saving time is not taken into account by the function block. You have to take appropriate measures in your application. Also refer to the *RTC Control / Timezone / Daylight Saving Time Example Guide* document *(see page 10)*.

The input `i_uiMaxRtcOffset` can be used as a plausibility check to verify the results from the server before the RTC of the controller is set.

NOTE: Setting the RTC of the controller generates entries into the controller log file. Therefore, for automatic adjustments, do not use this function more than once a day.



Interface

Input	Data type	Description
i_xEnable	BOOL	Activation and initialization of the function block.
i_xExecute	BOOL	The request specified with the input iq_stTimeRequest is executed upon rising edge of this input.
i_xSyncRtc	BOOL	If this input is set to TRUE, the RTC of the controller is set automatically after a response has been received from the (S)NTP time server. Default value: FALSE **NOTE:** Setting the RTC of the controller generates entries into the controller log file. Therefore, for automatic adjustments, do not use this function more than once a day.
i_diTimeZone	DINT	Offset that is added to the time stamp received from the (S)NTP time server in seconds. Default value: 0 Value range: -86400...+86400 (24 h x 60 min x 60 s)
i_uiMaxRtcOffset	UINT	Maximum difference allowed between the present and the new RTC value if i_xSyncRtc is TRUE. Can be used as a plausibility check to help prevent an unintended set of the RTC. For example, if the RTC is synchronized once with a time server, any following synchronizations (offsets) shall be in the range of ±30 s. Default value: 0 (function is inactive) Maximum value: 7200 s (2 hours)
i_stTimeRequest	ST_TimeRequest *(see page 32)*	Structure containing required user input data regarding the (S)NTP time server.

Output	Data type	Description
q_xActive	BOOL	If the function block is active, this output is set to TRUE.
q_xReady	BOOL	Indicates TRUE if the function block is ready to receive user input.
q_xBusy	BOOL	If this output is set to TRUE, the function block is processing.
q_xDone	BOOL	If this output is set to TRUE, the function block has successfully completed the requested command. Additional data can be collected from iq_stTimeRequest.
q_stTimeResponse	ST_TimeResponse *(see page 33)*	Structure containing the synchronized time stamp (including the offset i_diTimeZone), the clock offset between local RTC of the controller and the (S)NTP time server, and the round-trip delay.
q_Error	BOOL	If this output is set to TRUE, an error has been detected. For details, refer to q_etResult and q_etResultMsg.
q_etResult	ET_Result *(see page 29)*	Provides diagnostic and status information as a numeric value.
q_sResultMsg	STRING[255]	Provides additional diagnostic and status information as a text message.

Chapter 7
Functions

FC_EtResultToString

Overview

Type:	Function
Available as of:	V1.0.7.0
Inherits from:	–
Implements:	–

Task

Convert an enumeration element of type ET_Result to a variable of type STRING.

Functional Description

Using the function `FC_EtResultToString`, you can convert an enumeration element of type ET_Result to a variable of type STRING.

Interface

Input	Data type	Description
`i_etResult`	ET_Result	Enumeration with the result.

Return Value

Data type	Description
STRING(80)	The ET_Result converted to text.

Appendices

Appendix A
Function and Function Block Representation

Overview

Each function can be represented in the following languages:

- IL: Instruction List
- ST: Structured Text
- LD: Ladder Diagram
- FBD: Function Block Diagram
- CFC: Continuous Function Chart

This chapter provides functions and function blocks representation examples and explains how to use them for IL and ST languages.

What Is in This Chapter?

This chapter contains the following topics:

Differences Between a Function and a Function Block

Function

A function:

- is a POU (Program Organization Unit) that returns one immediate result.
- is directly called with its name (not through an instance).
- has no persistent state from one call to the other.
- can be used as an operand in other expressions.

Examples: boolean operators (`AND`), calculations, conversion (`BYTE_TO_INT`)

Function Block

A function block:

- is a POU (Program Organization Unit) that returns one or more outputs.
- needs to be called by an instance (function block copy with dedicated name and variables).
- each instance has a persistent state (outputs and internal variables) from one call to the other from a function block or a program.

Examples: timers, counters

In the example, `Timer_ON` is an instance of the function block `TON`:

```
PROGRAM MyProgram_ST
VAR
    Timer_ON: TON;  // Function Block Instance
    Timer_RunCd: BOOL;
    Timer_PresetValue: TIME := T#5S;
    Timer_Output: BOOL;
    Timer_ElapsedTime: TIME;
END_VAR
```

```
Timer_ON(
    IN:=Timer_RunCd,
    PT:=Timer_PresetValue,
    Q=>Timer_Output,
    ET=>Timer_ElapsedTime);
```

How to Use a Function or a Function Block in IL Language

General Information

This part explains how to implement a function and a function block in IL language.

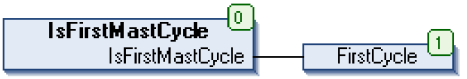
Functions `IsFirstMastCycle` and `SetRTCDrift` and Function Block `TON` are used as examples to show implementations.

Using a Function in IL Language

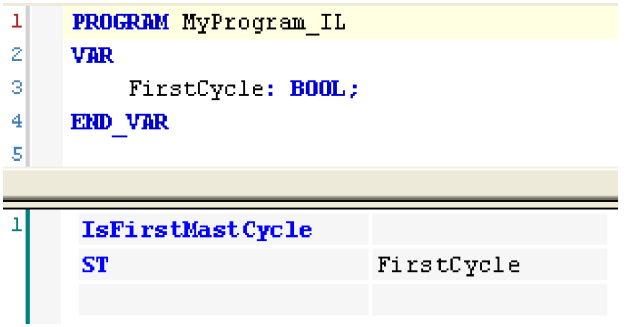
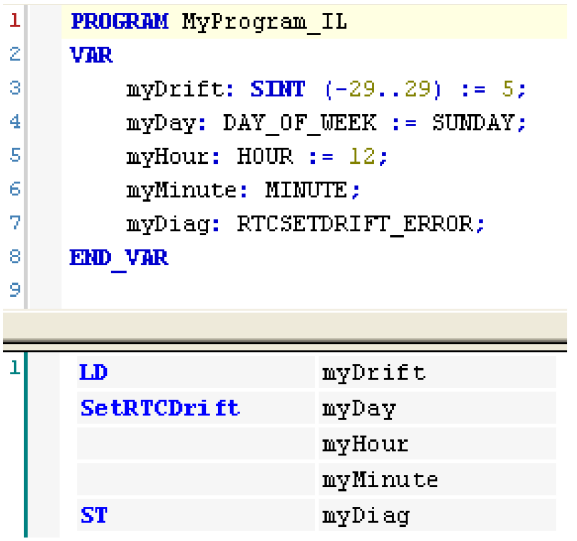
This procedure describes how to insert a function in IL language:

Step	Action
1	Open or create a new POU in Instruction List language. **NOTE:** The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs *(see SoMachine, Programming Guide)*.
2	Create the variables that the function requires.
3	If the function has 1 or more inputs, start loading the first input using LD instruction.
4	Insert a new line below and: ● type the name of the function in the operator column (left field), or ● use the **Input Assistant** to select the function (select **Insert Box** in the context menu).
5	If the function has more than 1 input and when Input Assistant is used, the necessary number of lines is automatically created with ??? in the fields on the right. Replace the ??? with the appropriate value or variable that corresponds to the order of inputs.
6	Insert a new line to store the result of the function into the appropriate variable: type ST instruction in the operator column (left field) and the variable name in the field on the right.

To illustrate the procedure, consider the Functions `IsFirstMastCycle` (without input parameter) and `SetRTCDrift` (with input parameters) graphically presented below:

Function	Graphical Representation
without input parameter: `IsFirstMastCycle`	
with input parameters: `SetRTCDrift`	

In IL language, the function name is used directly in the operator column:

Function	Representation in POU IL Editor
IL example of a function without input parameter: `IsFirstMastCycle`	
IL example of a function with input parameters: `SetRTCDrift`	

Using a Function Block in IL Language

This procedure describes how to insert a function block in IL language:

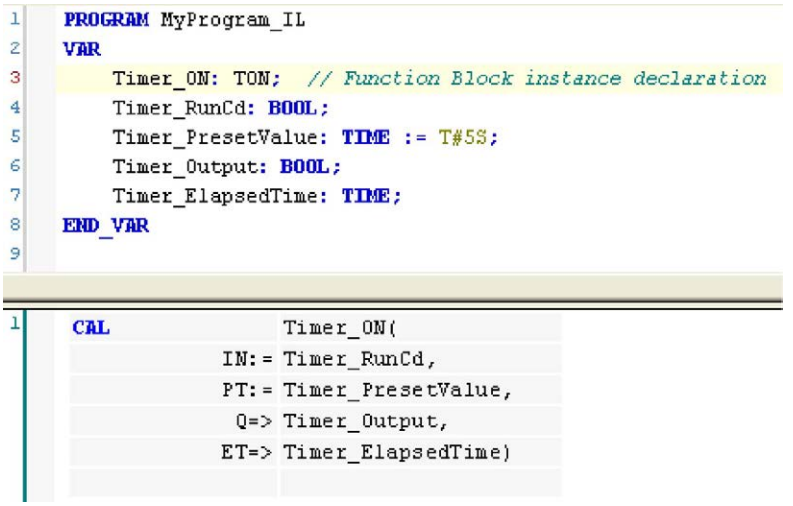
Step	Action
1	Open or create a new POU in Instruction List language. **NOTE:** The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs *(see SoMachine, Programming Guide)*.

Step	Action
2	Create the variables that the function block requires, including the instance name.
3	Function Blocks are called using a CAL instruction: ● Use the **Input Assistant** to select the FB (right-click and select **Insert Box** in the context menu). ● Automatically, the CAL instruction and the necessary I/O are created. Each parameter (I/O) is an instruction: ● Values to inputs are set by ":=". ● Values to outputs are set by "=>".
4	In the CAL right-side field, replace ??? with the instance name.
5	Replace other ??? with an appropriate variable or immediate value.

To illustrate the procedure, consider this example with the TON Function Block graphically presented below:

Function Block	Graphical Representation
TON	

In IL language, the function block name is used directly in the operator column:

Function Block	Representation in POU IL Editor
TON	

How to Use a Function or a Function Block in ST Language

General Information

This part explains how to implement a Function and a Function Block in ST language.

Function `SetRTCDrift` and Function Block `TON` are used as examples to show implementations.

Using a Function in ST Language

This procedure describes how to insert a function in ST language:

Step	Action
1	Open or create a new POU in Structured Text language. **NOTE:** The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs *(see SoMachine, Programming Guide)*.
2	Create the variables that the function requires.
3	Use the general syntax in the **POU ST Editor** for the ST language of a function. The general syntax is: `FunctionResult:= FunctionName(VarInput1, VarInput2,.. VarInputx);`

To illustrate the procedure, consider the function `SetRTCDrift` graphically presented below:

Function	Graphical Representation
`SetRTCDrift`	

The ST language of this function is the following:

Function	Representation in POU ST Editor

`SetRTCDrift`	`PROGRAM MyProgram_ST`
	`VAR myDrift: SINT(-29..29) := 5;`
	`myDay: DAY_OF_WEEK := SUNDAY;`
	`myHour: HOUR := 12;`
	`myMinute: MINUTE;`
	`myRTCAdjust: RTCDRIFT_ERROR;`
	`END_VAR`
	`myRTCAdjust:= SetRTCDrift(myDrift, myDay, myHour, myMinute);`

Using a Function Block in ST Language

This procedure describes how to insert a function block in ST language:

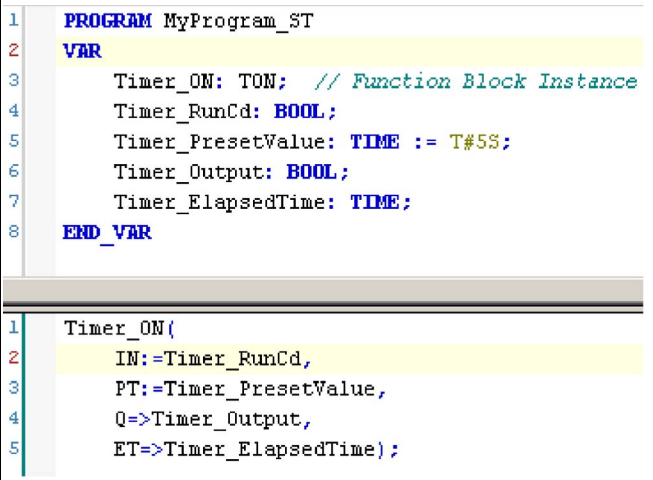
Step	Action
1	Open or create a new POU in Structured Text language. **NOTE:** The procedure to create a POU is not detailed here. For more information on adding, declaring and calling POUs, refer to the related documentation *(see SoMachine, Programming Guide)*.
2	Create the input and output variables and the instance required for the function block: ● Input variables are the input parameters required by the function block ● Output variables receive the value returned by the function block

3	Use the general syntax in the **POU ST Editor** for the ST language of a Function Block. The general syntax is:
	`FunctionBlock_InstanceName(Input1:=VarInput1, Input2:=VarInput2,...`
	`Ouput1=>VarOutput1, Ouput2=>VarOutput2,...);`

To illustrate the procedure, consider this example with the `TON` function block graphically presented below:

Function Block	Graphical Representation
`TON`	

This table shows examples of a function block call in ST language:

Function Block	Representation in POU ST Editor
`TON`	

Glossary

A

application

A program including configuration data, symbols, and documentation.

B

byte

A type that is encoded in an 8-bit format, ranging from 00 hex to FF hex.

C

CFC

(*continuous function chart*) A graphical programming language (an extension of the IEC 61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to the inputs of other blocks to create complex expressions.

configuration

The arrangement and interconnection of hardware components within a system and the hardware and software parameters that determine the operating characteristics of the system.

controller

Automates industrial processes (also known as programmable logic controller or programmable controller).

E

expansion bus

An electronic communication bus between expansion I/O modules and a controller.

F

FB

(*function block*) A convenient programming mechanism that consolidates a group of programming instructions to perform a specific and normalized action, such as speed control, interval control, or counting. A function block may comprise configuration data, a set of internal or external operating parameters and usually 1 or more data inputs and outputs.

function block diagram

One of the 5 languages for logic or control supported by the standard IEC 61131-3 for control systems. Function block diagram is a graphically oriented programming language. It works with a list of networks where each network contains a graphical structure of boxes and connection lines representing either a logical or arithmetic expression, the call of a function block, a jump, or a return instruction.

I

I/O

(*input/output*)

IL

(*instruction list*) A program written in the language that is composed of a series of text-based instructions executed sequentially by the controller. Each instruction includes a line number, an instruction code, and an operand (refer to IEC 61131-3).

INT

(*integer*) A whole number encoded in 16 bits.

L

LD

(*ladder diagram*) A graphical representation of the instructions of a controller program with symbols for contacts, coils, and blocks in a series of rungs executed sequentially by a controller (refer to IEC 61131-3).

P

POU

(*program organization unit*) A variable declaration in source code and a corresponding instruction set. POUs facilitate the modular re-use of software programs, functions, and function blocks. Once declared, POUs are available to one another.

program

The component of an application that consists of compiled source code capable of being installed in the memory of a logic controller.

R

RTC

(*real-time clock*) A battery-backed time-of-day and calender clock that operates continuously, even when the controller is not powered for the life of the battery.

S

SNTP

(*simple network time protocol*) A simplified version of the NTP (network time) protocol. It is used to synchronize computer clock times in a network of computers. It uses Coordinated Universal Time (UTC) to synchronize computer clock times to a millisecond, and sometimes to a fraction of a millisecond.

ST

(*structured text*) A language that includes complex statements and nested instructions (such as iteration loops, conditional executions, or functions). ST is compliant with IEC 61131-3.

U

UDP

(*user datagram protocol*) A connectionless mode protocol (defined by IETF RFC 768) in which messages are delivered in a datagram (data telegram) to a destination computer on an IP network. The UDP protocol is typically bundled with the Internet protocol. UDP/IP messages do not expect a response, and are therefore ideal for applications in which dropped packets do not require retransmission (such as streaming video and networks that demand real-time performance).

UDP

UDP (User Datagram Protocol) is a transmission protocol used in IP networks.

unicast / broadcast /multicast

Unicast communication targets one system as destination. For unicast communication, either TCP or UDP can be used.

Broadcast communication targets all systems in the same subnet. As with multicast, only UDP can be used.

Multicast targets some systems that have to join a specified multicast-group before being delivered packets targeted at that group.

UTC

(*universal time coordinated*) The primary time standard by which the world regulates clocks and time.

V

variable

A memory unit that is addressed and modified by a program.

Index